CSC 249/449 Deep Learning and Graphical Models: Homework 2

Term: Spring 2019
Instructor: Prof. Chenliang Xu
TA: Yapeng Tian, Jing Shi, Zhiheng Li, Yutong (Kelly) He, Chaoying Xue
Due Date: 11:59 pm, 02/16/2019

Problem 1 (5 pts): Negative Log Likelihood Loss
A negative log likelihood loss is defined as follows:

$$\begin{aligned} L &= -\log(p_{gt}), \\ p_{gt} &= \text{softmax}(\mathbf{y}, gt) \\ &= \frac{\exp(y_{gt})}{\sum_{i=1}^{n} \exp(y_i)}, \end{aligned}$$

where $\mathbf{y} = [y_1, y_2, \ldots, y_n]^T$ and gt is a scalar that indicates the ground truth label. Prove that

$$\frac{\partial L}{\partial y_i} = \begin{cases} p_i - 1, i = gt, \\ p_i, i \neq gt. \end{cases}$$

Problem 2 (5 pts): Backpropagation of Fully-Connected Layer
In fully-connected layer, an input feature vector $\mathbf{x}$ with L elements ($\mathbf{x} \in \mathbb{R}^{L\times 1}$) will multiply with a weight matrix $\mathbf{W} \in \mathbb{R}^{O\times L}$ plus a bias term $\mathbf{b} \in \mathbb{R}^{O\times 1}$, which is defined by:

$$\mathbf{y} = \mathbf{W}\mathbf{x} + \mathbf{b},$$

where $\mathbf{y}$ is the output of the fully-connected layer. Assume $\frac{\partial L}{\partial \mathbf{y}}$ is given, what is the derivative of the loss L with respect to $\mathbf{W}$, $\mathbf{b}$ and $\mathbf{x}$, respectively? (For full credit, pleases provide details about how you derive your answers.)

Problem 3 (10 pts): Backpropagation of 2D Convolutional Layer
Given an input feature map $\mathbf{I}$ with T channels, a 2D convolution with K filters of size $T \times M \times N$ at stride s is defined as

$$y(k,i,j) = \sum_{t=0}^{T-1}\sum_{m=0}^{M-1}\sum_{n=0}^{N-1} x(t, i\times s + m, j \times s + n) W_k(t,m,n) + b_k,$$

where W_k and b_k are the filter and bias of kth neuron. Assume $\frac{\partial L}{\partial \mathbf{y}}$ is given, what is the derivative of the loss L with respect to $W_k(t,m,n)$, b_k and $x(t,m,n)$, respectively? (For full credit, pleases provide details about how you derive your answers.)

Problem 4 (10 pts): Backpropagation of 2D Max Pooling Layer
Given an input feature map $\mathbf{I}$ with C channels, then a 2D max pooling filter of size $C \times M \times N$ at stride s is defined as

$$y(c,i,j) = \underset{m=0..M-1}{\text{Max}} \; \underset{n=0..N-1}{\text{Max}} \; x(c, i\times s + m, j\times s + n),$$

where c is the index of channel and Max outputs the maximum value from the sequence. Assume $\frac{\partial L}{\partial \mathbf{y}}$ is given, what is the derivative of the loss L with respect to $x(t,m,n)$? (For full credit, pleases provide details about how you derive your answers.)

Problem 5 (70 pts): Implementing Backpropagation of 2D Convolutional Layer and Max Pooling Layer

In this homework you will work with a framework similar to problem 2 we provide you in Homework 1. Following the last assignment in which you have implemented the forward computation of convolutional layer and max pooling, you are going to implement backward pass of those two layers this time.

Do **NOT** modify function interfaces. If you want to add parameters to a function, please provide default values so that the original behavior of the function is unchanged.

Except the functions marked with ”TODO”s, do **NOT** make any changes to the framework (e.g. the Graph class). You should be able to finish this assignment with the provided utility functions and classes.

Do **NOT** use any additional python packages/modules other than those provided in the framework.

1. (10 pts) Fully-Connected Layer In this assignment, the fully-connected layer multiplies the input vector with a matrix matrix and then adds a bias term.
   - Implement the *backward* passes of FullyConnected layer in the “layer” module (See **TODO** comments in *layer.py*).
   - Pass the unit test in *test_backprop.py*.
2. (40 pts) Convolutional Layer

   The convolutional layer you will implement in this assignment will work for input feature maps with channels. That is, the input feature maps will be 3D of shape (C, H, W) where C denotes the number of channels. The convolutional layer should support stride and padding and have multiple neurons, i.e. output multiple activation maps.

   Here (https://arxiv.org/abs/1603.07285) is a very good technique report about implementing convolution layers. Especially, you may refer Section 2.4 and Section 4.6, Relation 13 for implementing the forward and backward passes, respectively.

   Similar with homework 1, we are seeking a direct implementation of convolution with nested for loops in backward pass.
   - Implement the *backward* passes of Conv2D layer in the “layer” module (See **TODO** comments in *layer.py*).
   - Pass the unit test in *test_backprop.py*.
3. (20 pts) Max-Pooling Layer

   The max pooling layer you will implement in this assignment will work for input feature maps with channels. That is, the input feature maps will be 3D of shape (C, H, W) where C denotes the number of channels.

   Note that maximum value may not be unique during the forward pass. Therefore, the backward pass of max pooling should assign the gradient to multiple activations which achieve the same maximum value.
   - Implement the *backward* passes MaxPool2D layer in the “layer” module (See **TODO** comments in *layer.py*).
   - Pass the unit test in *test_backprop.py*.

Submission Process: You should prepare your submission using the *collect_submission.py* program. Please run the program and upload the generated zip file to Blackboard. Your submission should contain the following:

- code/ - The implementation of the framework we provide you.
- homework.pdf - Your answers to Problem 1, 2, 3, and 4.

Grading and Evaluation: The credit for each problem in this set is given in parentheses at the stated question (sub-question fraction of points is also given at the sub-questions). Partial credit will be given when appropriate.